Deferred decentralized movement pattern mining for geosensor networks

Laube, P ; Duckham, M ; Palaniswami, M

Abstract: This paper presents an algorithm for decentralized (in-network) data mining of the movement pattern flock amongst mobile geosensor nodes. The algorithm DDIG (Deferred Decentralized Information Grazing) allows roaming sensor nodes to 'graze' over time more information than they could access through their spatially limited perception range alone. The algorithm requires an intrinsic temporal deferral for pattern mining, as sensor nodes must be enabled to collect, memorize, exchange, and integrate their own and their neighbors' most current movement history before reasoning about patterns. A first set of experiments with trajectories of simulated agents showed that the algorithm accuracy increases with growing deferral. A second set of experiments with trajectories of actual tracked livestock reveals some of the shortcomings of the conceptual flocking model underlying DDIG in the context of a smart farming application. Finally, the experiments underline the general conclusion that decentralization in spatial computing can result in imperfect, yet useful knowledge.

DOI: https://doi.org/10.1080/13658810903296630

research article

Deferred Decentralized Movement Pattern Mining for Geosensor Networks

Patrick Laube[a]*, Matt Duckham[a], and Marimuthu Palaniswami[b]

[a] *Geomatics Department, The University of Melbourne, Parkville 3010 VIC, Australia*;
[b] *Department of Electrical and Electronic Engineering, The University of Melbourne, Parkville 3010 VIC, Australia*

(*received yesterday, accepted today, published tomorrow*)

This paper presents an algorithm for decentralized (in-network) data mining of the movement pattern *flock* amongst mobile geosensor nodes. The algorithm `DDIG` (Deferred Decentralized Information Grazing) allows roaming sensor nodes to 'graze' over time more information than they could access through their spatially limited perception range alone. The algorithm requires an intrinsic temporal deferral for pattern mining, as sensor nodes must be enabled to collect, memorize, exchange, and integrate their own and their neighbors' most current movement history before reasoning about patterns. A first set of experiments with trajectories of simulated agents showed that the algorithm accuracy increases with growing deferral. A second set of experiments with trajectories of actual tracked livestock reveals some of the shortcomings of the conceptual flocking model underlying `DDIG` in the context of a smart farming application. Finally, the experiments underline the general conclusion that decentralization in spatial computing can result in imperfect, yet useful knowledge.

Keywords: Geosensor networks, decentralized spatial computing, trajectory data mining, movement patterns, flocking

1. Introduction

Decentralized spatial computing (DeSC) aims at generating global knowledge using local information processing (cf. Estrin *et al.* 2000). The increasing use of highly distributed spatial information systems, such as geosensor networks (wireless networks of miniaturized sensor-enabled computers monitoring phenomena in geographic space, Nittel *et al.* 2004), is driving increased research and application of decentralization of spatial computing. This paper presents a DeSC approach for data mining the movement pattern *flock* amongst mobile geosensor nodes. A flock is defined as a set of mobile entities that maintain spatial proximity over an extended period of time (see Figure 1).

Movement patterns such as *flocking* (Laube *et al.* 2004) or *leading* (Andersson *et al.* 2008) represent high-level process-knowledge derived from low-level trajectory data. Knowledge about coordination amongst moving entities is essential to various application fields. For example, in traffic management flocking vehicles may indicate congestion or traffic jams; firefighters battling a bush-fire in poor visibility must maintain spatial proximity and movement coordination; in 'smart farming' applications, flocking sheep may need to be dispersed in order to reduce nitrogen

*Corresponding author. Patrick is currently affiliated with the Department of Geography, The University of Zurich, email: patrick.laube@geo.uzh.ch

leaching (Betteridge 2008) and bulls may need to be kept apart to avoid fights (Wark *et al.* 2007). For this study, flocking animals not only provide the inspirational metaphor for defining the generic movement pattern *flock*, but tracking data of herding cattle in confinement is also used for evaluating the decentralized data mining algorithms developed.

Movement patterns have recently received a lot of attention in GIScience, database research, and computational geometry. Apart from conceptual work defining movement patterns (Laube and Imfeld 2002), most research has focused on the efficient detection of flocks in centralized database systems (Andersson *et al.* 2008, Gudmundsson *et al.* 2007, Laube *et al.* 2004). Only recently, initial work was presented aiming at decentralized detection of movement patterns in a mobile wireless sensor context (Laube *et al.* 2008). The present paper reports on the progression of this research, introducing (i) a more efficient flock detection algorithm based on self-localizing sensor nodes and exploiting spatiotemporal memory, and (ii) empirical evaluation of algorithm performance on simulated data and on real animal trajectories.

The paper presents the algorithm `DDIG` (Deferred Decentralized Information Grazing). In this context *information grazing* refers to sensor nodes that gather information whilst roaming. Sensor nodes extend their knowledge beyond the limits of their spatially limited perception range by exploiting time. Hence `DDIG` is *deferred* as it allows the sensor nodes to first collect information, only after a deferral addressing the actual pattern detection task. Finally, `DDIG` is also *decentralized* as the sensor nodes achieve their goal through local in-network collaboration by exchanging and integrating locally collected data. No global data processing or centralized control is required for mining the flock patterns.

Following a critical review of related work in Section 2 and a formal problem definition in Section 3, the decentralized `DDIG` algorithm is introduced in Section 4. The algorithm performance is evaluated using correlated random walk simulations (Section 5) and real cattle trajectory data (Section 6). Finally, an analysis of the experimental results and their implications with respect to formalizing and mining flocks is discussed in Section 7 with conclusions for future work in Section 8.

2. Related Work

Wireless sensor networks (WSN) are wireless ad-hoc networks of untethered, battery powered, and sensor-enabled miniature computing platforms (Zhao and Guibas 2004). WSNs allow close monitoring of natural and built spaces, and are particularly suited to inaccessible and even hostile environments. WSNs monitor the bleaching of the coral reef (Chatterjea *et al.* 2006), observe seismic activity (Werner-Allen *et al.* 2006), or follow traffic flow (Kellerer *et al.* 2001). WSN are especially suited for monitoring dynamics in geographic spaces. Such *geosensor networks* (Nittel *et al.* 2004) offer a 'close sensing' complement to conventional remote

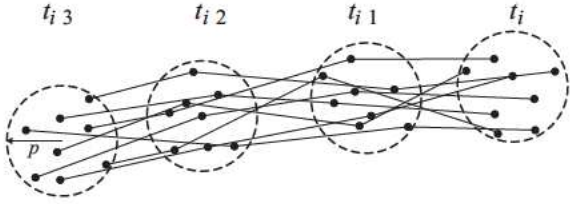

Figure 1. Eight moving entities building a flock from t_i to t_{i-3}.

sensing, potentially covering previously impossible spatial and temporal granularities of environmental monitoring. The majority of research in this technology-driven area has been focused on the establishment and maintenance of the network infrastructure (e.g. Braginsky and Estrin 2002, Cheng and Heinzelman 2005). Recent research on geosensor networks, however, assumes the existence of technical networking capabilities, and explicitly addresses the spatial application layer. Such spatial applications of WSN include the detection of dynamics in fields (Duckham *et al.* 2005), topological change (Farah *et al.* 2008, Sadek and Duckham 2008) and movement patterns as in the present paper.

In the context of spatial applications, *self-localization* (determining the geographic position of nodes) can be a crucial sensing task for a WSN. For example, localization is basic to *geographic routing*, where information packets are forwarded from sensor node to sensor node, at each 'hop' minimizing the distance to target node (Karp and Kung 2000). Although GPS provides one possible positioning technology, many other positioning technologies are available and more suited to low-power, low-cost geosensor networks (such as WiFi triangulation, RF or ultrasound range- and direction-finding, Zhao and Guibas 2004). Whereas localization may be part of the initialization of a static network, in *mobile wireless sensor networks* (mWSN, Laube *et al.* 2008), a node's position constantly changes and needs to be constantly updated. Previous work on the decentralized detection of flocking patterns did not imply position knowledge (Laube *et al.* 2008), this paper by contrast explicitly aims at exploiting the advantage of sensor nodes knowing their coordinate position.

Most conventional spatial computing and spatial data mining techniques adopt a centralized architecture, where information from a network is collated and processed centrally. Specifically in the content of this paper, algorithms for data mining of spatial movement pattern have been developed that use global data structures (for *flocking* in Gudmundsson *et al.* 2007) or preprocessed metadata arrays (for *leadership* in Andersson *et al.* 2008). However, WSNs and mWSNs fundamentally change the way spatial data is captured and processed. Conventional spatial computing requires all data be collated in central data processing unit. In a WSN that is typically not possible, as such data transfer would clog the network's communication bandwidth and deplete the nodes batteries. As a result, research is increasingly focusing on distributing computing tasks throughout the network, especially important in the data rich spatial domain. The term *decentralized* is used to refer specifically to a distributed system where no component knows the entire system state (Lynch 1996). In decentralized systems, individual elements must cooperate to complete some processing task, but both the task and the data remain distributed throughout the network. Hence, *decentralized spatial computing* aims at the development of algorithms that can operate using purely *local* knowledge, but are still able to monitor geographic phenomena with *global* extents (Laube and Duckham 2009, Estrin *et al.* 2000).

This paper aims to develop a decentralized spatial algorithm for detecting movement patterns, more suited to use in a highly dynamic, highly distributed spatial computing system, like an mWSN. The algorithm adapts a conventional filter-refine paradigm from databases (Brinkhoff *et al.* 1994, Becker *et al.* 1999, Esperanca and Samet 1997), increasing algorithm efficiency by spatially and temporally filtering unnecessary data before performing potentially more expensive spatial computation on filtered data.

3. Decentralized Detection of Flocks in a mWSN

Before presenting the DDIG algorithm, this section sets out more precisely the underlying assumptions and context for decentralized detection of flocks in an mWSN.

For a definition of flocking we follow Benkert *et al.* (2008) and define an (n, k, p)-*flock* as any set M of n mobile sensor nodes, where for every moment in a time period of k consecutive time steps, there exists some disk of flock radius p that contains every sensor node in M (see Figure 2a). n shall be termed *flock cardinality*, k termed *flock contiguity*, and p termed *flock radius*.

An mWSN is defined as a set A of mobile sensor nodes. The movement trajectories of sensor nodes in the plane ($\mathbb{R}^2$) can be modeled as a *locator* function $l : A \times T \rightarrow \mathbb{R}^2$, where $T = \{t_1, t_2, ..., t_n\}$ is an ordered set of discrete time steps. Thus for any time $t \in T$ and sensor node $a \in A$, $l(a, t)$ gives the coordinate location of sensor node a at time t. Note that the localization of the sensor nodes is one major difference to earlier work, especially to the qualitative decentralized flock detector FLAGS (Laube *et al.* 2008), where sensor nodes have no localization capabilities and rely on purely qualitative information about sensor neighborhoods. In the context of our definition of mWSN, an (n, k, p)-*flock* is a set $M \subseteq A$ such that for all times in a period $\{t_i, t_{i+1}, ...t_{i+k}\} \subseteq T$ there exists a circle of radius p that spatially contains $l(m, t)$ for all $m \in M$.

Communication between sensor nodes in a WSN or mWSN is restricted by limited power resources. As a result, a pair of sensor nodes $a, a' \in A$ can only communicate directly with one another at a particular time t if they are within communication range c. i.e., $\delta(l(a, t), l(a't)) \leq c$ where $\delta(a, b)$ is the usual Euclidean metric for distance between two points. For simplicity the present model assumes all sensor nodes have constant and equal communication distances (see Figure 2b).

The communication range c serves as a critical constraint on a sensor node's knowledge about space, and is analogous to a sensor node's 'perception' range. The communication range of a node typically encompasses only a small part of the space (i.e., for a node a at time t, $|\{a' \in A|\delta(l(a, t), l(a't)) \leq c\}| \ll |A|$). The challenge for decentralized flock detection occurs when the communication range c drops below flock radius p (otherwise some sensor nodes will normally be able to perceive a flock in its entirety, see Figure 2c). To address this problem, the DDIG algorithm exploits the spatiotemporal characteristics of movement, allowing the sensor nodes to graze information in space-time and learn about their environment beyond their limited communication range.

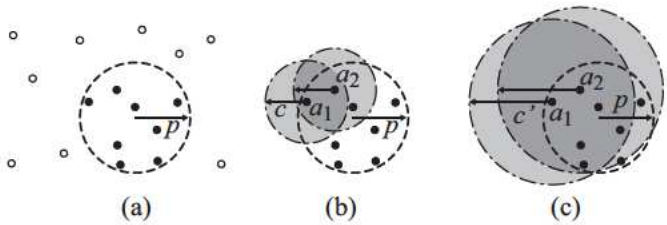

Figure 2. Problem definition. (a) Set A of localized sensor nodes a, of which $n = 8$ are flocking within flock radius p (only one time step t_0 illustrated, flocking sensor nodes in black, non-flocking nodes in white). (b) Nodes within communication range $c = 0.75p$ can exchange information, for example a_1 and a_2. (c) The ratio $\frac{c}{p}$ determines the fraction of a pattern an individual could possibly perceive itself, with $c = 1.25p$ sensor node a_2 'sees' all neighbors of the flock.

4. The DDIG algorithm

This section presents the **D**eferred **D**ecentralized **I**nformation **G**razing (DDIG) algorithm, allowing the decentralized data mining of flock patterns in a mobile WSN. DDIG is based on two core concepts. First, it equips sensor nodes with spatiotemporal memory such that they can recall their recent movement history. This memory allows the mobile sensor nodes to gather information before the *deferred* data processing starts. Second, a *decentralized*, spatiotemporal adaptation of the common spatial database 'filter-refine' strategy for query optimization allows the sensor nodes to exchange the gathered location data with their neighbors and condense the thereby gained information to knowledge about movement patterns. In combination, these two strategies enable sensor nodes to *graze information* over time.

The algorithm has three main steps. At each timestep each sensor node:

(1) maintains a data structure updated with new information about the most recent movements of that node ('tails' step, Section 4.1);
(2) communicates information about its tail with its neighbors, subsequently stored in the neighbors' memories ('filter' step, Section 4.2); and
(3) processes all locally stored tails in memory to identify any flocks ('refine' step, Section 4.3).

Each of these steps is explored in more detail in the following subsections.

4.1. *Tails*

Information about a sensor node's recent trajectory is termed a 'tail'. Sensor nodes could potentially store information about all their previously visited locations. However, such a strategy would place a strain upon the limited memory capacity of low-cost sensor nodes. Further, since sensor nodes are constantly processing stored information to detect flocks, older information becomes decreasingly relevant to this analysis. Consequently, in the DDIG algorithm, nodes store tails as a maximum sized queue (FIFO, first in first out) of length e. For example, at a particular time step t_i, a sensor node a_1 in Figure 3a with tail length $e = 7$ stores in its current tail the $e - 1 = 6$ previously visited position *fixes* at times t_{i-1} until t_{i-6}. In effect, a sensor node's memory degrades over time, with older knowledge about location being forgotten. The minimal tail length e for flock detection is equal to flock contiguity k, in order to cover at least all past time steps required for a flock. However, in order to allow for additional information grazing, the tail can be extended by a given *deferral* d, hence $e = d + k$.

4.2. *Filter*

Conventional spatial databases commonly adopt the *filter-refine* paradigm to improve efficiency of spatial query processing. When responding to a spatial query, a spatial database will typically first discard any data that cannot possibly satisfy the query (*filter*), before performing the computationally expensive spatial processing or analysis on the filtered data set (*refine*). Because spatial algorithms are typically computationally expensive the approach is critical to efficient and rapid query processing in spatial databases. In a similar way, the DDIG algorithm performs an analogous, decentralized version of filter-refine. In DDIG nodes exchange information only with spatially nearby nodes. Since a flock is a local pattern, restricting information exchange and collaboration to a node's local neighborhood is a reasonable filter for DDIG. Such spatial filtering ensures that information ex-

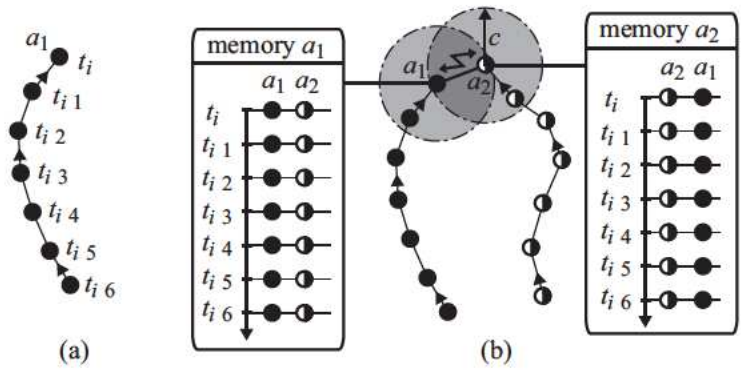

Figure 3. (a) Sensor node a_1 carries a memory tail of extent $e = 7$, memorizing the past position fixes for times t_i until t_{i-6}. (b) When connected, nearby nodes a_1 and a_2 exchange tails and store their own and the received position fixes in their memory.

change is restricted to nodes in the neighborhood where it is most relevant, thereby reducing use of limited computational and communication resources.

The tails are the information packages that are exchanged locally. The key question for the filter step, then, is to define the rules which determine what is meant by 'local', that is which tails a sensor node receives. Given that nodes are searching for a flock with a radius of p, in order not to miss a potential partner building a flock of radius p a sensor node must filter as far out as $2 * p$. This becomes obvious when considering a sensor node sitting at the edge of a flock (such as for example a_2 in Figure 4a, needing to poll tails as far away as a_5). Therefore, in the filter step every sensor node queries all other nodes positioned within a disc of radius $2 * p$ for their own tails. Since sending and receiving sensor nodes both know their positions, the query can be spatially limited with a query range constraint added to the query message. Since $2 * p$ may easily exceed communication range c, this pull query may require multiple hops. Intrinsically broad filtering will inevitably query tails of sensor nodes that are not part of a flock, as is illustrated in Figure 4c for a larger communication range $c' = p$. Subsequent *refine* will dispose of unneeded tails.

However, at a given time t_i only connected sensor nodes within $2 * p$ will provide their tails. This means that for unfavorable constellations sensor nodes building a flock may not be connected and hence not be able to exchange their knowledge.

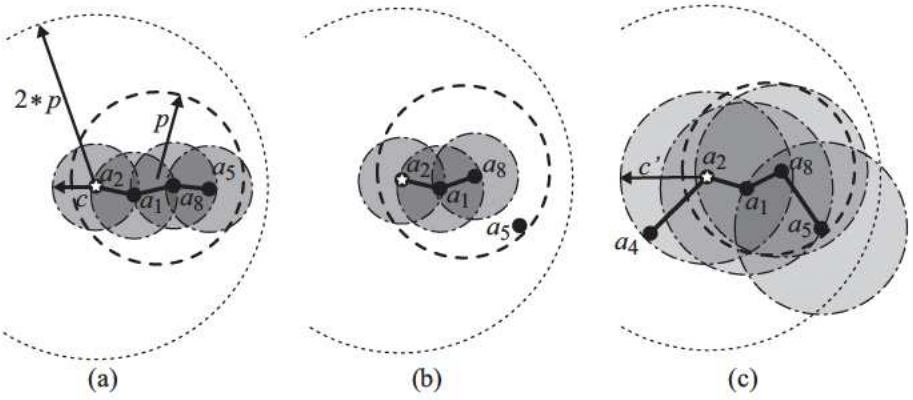

Figure 4. Spatial filter. (a) a_2 grabs in the filter step through multi-hop all the tails of other sensor nodes it is connected through communication range $c = 0.5 * p$ and that are closer than $2 * p$. Even though located within flock radius p, in an alternate constellation sensor node a_5 is not included as it fails the 'connected' condition. (c) With a larger communication range $c = p$, however, a_2 receives additionally the tail of non-flocking node a_4 and has to rely on the subsequent refine step dropping the superfluous candidate.

For instance in Figure 4b, given communication range $c = 0.5 * p$, sensor node a_5 is not connected to a_2 and will not be able to deliver its tail to a_2s memory at time t_i. DDIG-filter hence deviates from conventional spatial filtering as it can't guarantee not to miss candidates. However, here the deferred processing of DDIG comes into play, as this missing communication link can be closed in one of the next time steps and a_5 can again contribute to the aggregate knowledge of a_2.

Consider, for example, the situation depicted in Figure 5 at t_{i-4}: Even though node a_2 is flocking with a_1, a_4, a_5, and a_8, at this very time a_2 is only connected through communication range c (grey disc) to a_4. Here, spatial filtering alone can't guarantee to pick up all flocking candidates. However, at subsequent time t_{i-3}, a_2 is connected with a_1 and a_8, then receiving information about those tails that were previously missed out. Equally, a_2 receives a_5's tail at t_{i-1}. In other words, the spatiotemporal movement of nodes is exploited over time in order to compensate for poor connectivity and spatial configurations at any particular time instant. Nevertheless, filtering in DDIG cannot guarantee that a favorable constellation of nodes will eventuate. Thus, DDIG filtering is analogous, but not equivalent, to filtering in databases.

Each node aggregates the tails it receives at t_i in a local rolling memory, again of a fixed length covering the past e time steps (Figure 3b). With the tails, a node receives its neighbors' previously visited locations and stores them in memory. Even if two sensor nodes are not connected anymore at t_i, in their respective memories they can still have observation fixes of each other (except the most recent ones). For example, in Figure 5 sensor node a_2 receives tails from nodes a_4 at t_{i-4}, a_8 at t_{i-3} and a_5 at t_{i-1}. This knowledge persists for some time in memory,

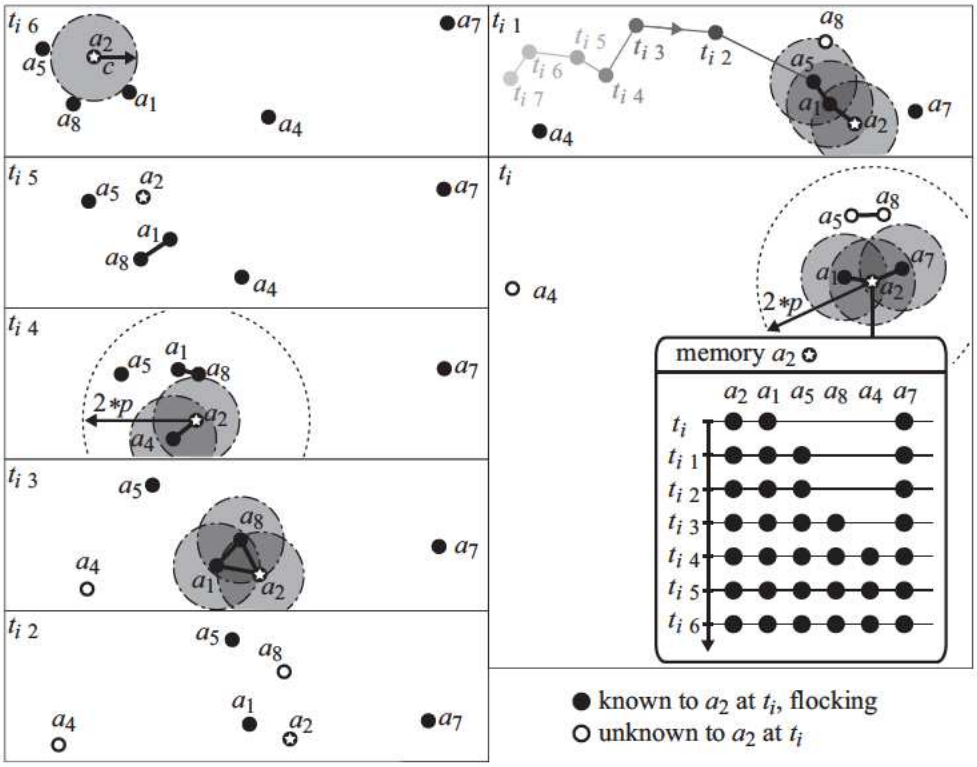

Figure 5. Temporal filter. DDIG builds a memory of the current movement history of nearby nodes. Example sensor node a_2 with communication range c roams from t_{i-6} to t_i and encounters nearby nodes. Whenever a_2 is connected to nearby nodes within $2 * p$ ($2 * p$-disc is only illustrated for t_i), it pulls their tails. a_2 received tails from a_4 at t_{i-4}, from a_8 at t_{t-3}, from a_5 at t_{i-1} (entire tail illustrated for t_{i-2}), and finally from a_1 and a_7 at t_i. Memory a_2 reflects the corresponding build-up of local knowledge. Note, a_2 at t_i remembers previous positions of nodes it is currently not connected to (a_4, a_5, and a_8).

even if a_2 subsequently becomes disconnected from these nodes at t_i. In this way, roaming sensor nodes graze knowledge about their neighbors over time: whereas a_2 only knows about three neighbors at t_i, it has accumulated knowledge about six neighbors at t_{i-6}.

Note that the filter step in `DDIG` is in fact not only spatial, but spatiotemporal. The filter step not only searches space at current time t_i but also reaches back in time. The algorithm allows the exchange of information about previous fixes, for 'retrospective' processing in the deferred refine step.

`DDIG` is designed to exploit the roaming behavior of sensor nodes as this may result in filling in knowledge gaps that appear when the communication graph becomes disconnected for small communication ranges c. Clearly, there is no guarantee that all gaps are filled and `DDIG` is expected to become error-prone for very small cs. Similarly, filtering with $2 * p$ results in querying nodes receiving tails of non-flocking nodes. This does not results in errors as the refine step discards superfluous information.

4.3. *Refine*

After having received tails of potential flocking neighbors, each sensor node has built up extended spatiotemporal knowledge about its neighborhood and several algorithmic approaches could now be taken for the refine step—confirming or rejecting the actual presence of a flock. A set of heuristics and approximations have been suggested for the optimization of related problems. Benkert *et al.* (2008) presented several efficient centralized algorithms for approximating the flock detection problem for short flock contiguities (small k). Similarly, Laube *et al.* (2008) presented a decentralized algorithm based on a local extrapolation heuristic, and an analysis of the inevitable errors of omission and/or commission associated with any heuristic. In both cases the results of the presented algorithms depend very much on the characteristics of the input point sets and there is no single best solution on offer. Since our goal was not the optimization of the refine step in memory *per se*, but rather the presentation of the deferred processing as a concept for DeSC, we did not engage in this important discussion. Instead, we used a simple heuristic based on outlier elimination that suited the memory data structure.

For each time step in memory, the median center of all fixes is computed. The heuristic now assumes that the flocking fixes are located within a disc of roughly flock radius p around the median center. A refine width w slightly larger than p is chosen and fixes beyond that refine width around the median center are dropped for each time step, resulting in the set of ids of potentially flocking neighbors per time step. Neighbor id-sets that persist in memory for more than k consecutive time steps build a flock.

This heuristic applied to a memory of extent $e = d + k$ allows the detection of flocks of contiguity k for different times, ranging from t_i to t_{i-d}. However, this paper aims to illustrate that deferred processing can be exploited in a dynamic decentralized system. For this reason `DDIG` was configured to always allow for maximal deferral and only detect flocks for the last possible time t_{i-d} (Figure 6). A flock is detected for $t - d$ if the intersection of the id-sets for the last k time steps in memory is equal or larger than flock cardinality n.

In the used heuristic, the refining width w allows two forms of detection error to be balanced. When w is chosen too narrow ($w \approx p$), `DDIG` may miss out on some flocks. When w is chosen too wide ($w \gg p$), false positive results may emerge.

5. Evaluation with simulated data: Correlated random walk

The second main contribution of this paper consists of a thorough experimental evaluation of the decentralized flock detector DDIG. In this section, simulated movement data were used for testing reliability of the algorithm. In Section 6, trajectories of real gazing cattle were used for testing the validity of the formal flocking model in a real application scenario. Both test environments were implemented using the open source agent-based simulation toolkit REPAST.

Previous work evaluating flock detection algorithms used simulated trajectories based on unconstrained random movement (URM) (Benkert *et al.* 2008) and both URM and correlated random walk (CRW) (Laube *et al.* 2008). However, Laube *et al.* (2008) concluded that in order realistically to test algorithms for mining structure in trajectory data, structured CRW data is preferred to unstructured URM data. Specifically, the exploitation of the spatiotemporal characteristics of movement requires trajectories that express spatiotemporal autocorrelation or *locality* as in CRW. Locality refers to the property of CRW where consecutive fixes are in proximity, simulating the progressive path of an animal or a pedestrian. CRW provides such locality as it is based on consecutive steps with step lengths and turning angles drawn from some stochastic frequency distribution. Hence, the first evaluation in this paper is based on CRW only.

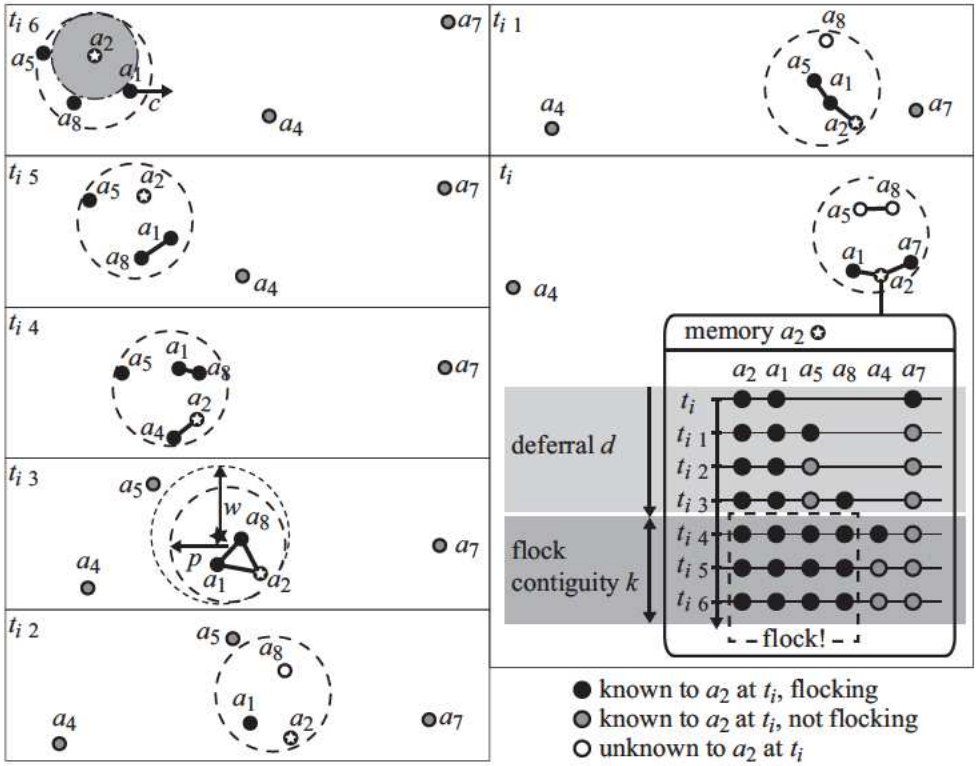

Figure 6. Refine. The refine step computes for all time steps in memory $[t_{i-6}, \ldots, t_i]$ the id-set of flocking sensor nodes that lie within refine width w around the median center (star, see t_{i-3}). If the intersection of the id-sets found in the oldest k time steps in memory is equal or larger flock cardinality n, a flock is detected for $[t_{i-6}, \ldots, t_{i-4}]$. Note that, given the deferred processing and the information stored in memory, a_2 finds flocks even with neighbors it is not connected anymore at t_i.

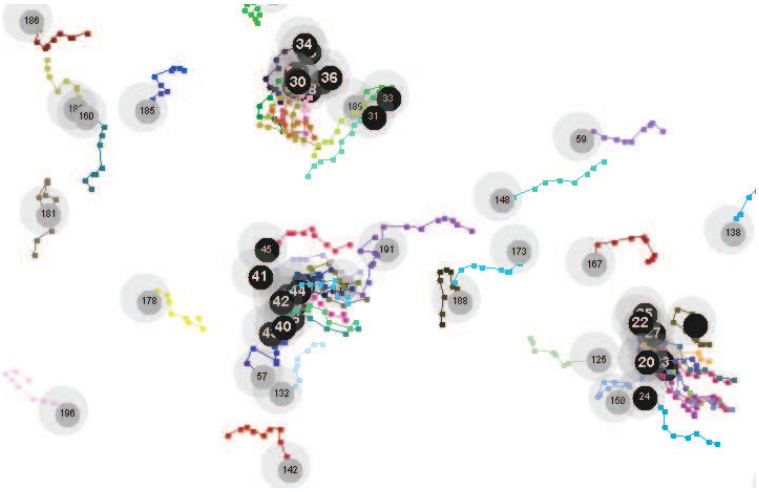

Figure 7. Enlarged snapshot of the map view of the repast simulation environment. The section shows three flocks of $n = 10$ flocking sensor agents (black, white letters) and several non-flocking sensor agents (gray, black letters) with their memory tails and communication range $c = 0.5 * p$. Flocking sensor agents with bold large letters have detected the flock (for example a_{30}, a_{34}), those with small letters haven't (for example a_{31}, a_{32}).

5.1. *Data*

Reliability testing was performed on a scenario featuring flocking and non-flocking sensor agents altogether moving in a confined space. Since at any time the number and the properties of the flocks in the simulation were known, DDIGs performance could be evaluated by the fraction of patterns found or missed compared to the patterns present.

For all experiments a population of 200 sensor nodes ($|A| = 200$) was simulated moving over $|T| = 100$ discrete time steps in a square space of size 8192^2 positions. The sensor nodes' movement was simulated as CRW using a normal distribution with a direction change $N(\mu = 0, \sigma = 1.0\pi)$ and a step length of $N(\mu = 50, \sigma = 25)$ was used. 50 out of 200 sensor nodes were given flocking behavior, implanted in 5 flocks consisting of $n = 10$ flocking sensor nodes each with a flock radius $p = 250$. See Figure 7 for a detailed snapshot view of the REPAST map view featuring flocking and non-flocking sensor agents.

5.2. *Methodology and preliminaries*

The evaluation of DDIG was conducted as an error analysis. For testing the reliability of the algorithm, at each time step the number of found patterns was compared with the known number of implanted patterns. Two forms of error were investigated: First, *errors of omission*, or the number of implanted patterns that were not detected (*eoo*, 'missed patterns'); Second, *errors of commission*, or the number of wrongly detected patterns when no pattern was implanted in the first place (*eoc*, 'false positives'). Errors were evaluated and averaged over all $|T| = 100$ time steps and a sample of 5 simulation runs per experiment.

5.3. *Experiment #1: Deferred processing*

The key challenge in decentralizing movement pattern mining in a geosensor context is delivering acceptable reliability when the communication range c of individual sensor nodes drops below the spatial extent of the pattern to be detected, here the flocks of radius p (illustrated in figure 2b). With a $\frac{c}{p} - ratio$ between

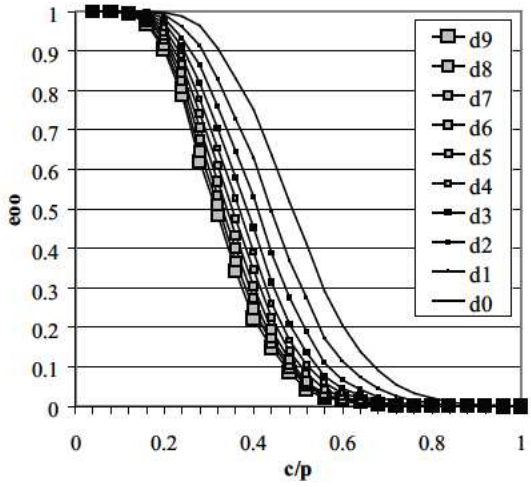

Figure 8. Deferred processing. The figure shows the error of omission (*eoo*, 'missed patterns') for a deferral varied from $d = 0$ to $d = 9$ for a flock contiguity $k = 3$. Longer deferrals and hence larger memories are represented with larger square symbols. As a main result, the experiment shows that *eoo* is reduced with prolonged deferrals. This effect is prominent for short deferrals and thins out with growing deferrals. Comb width $w = 1.2$, partial pattern size $n' = 8$.

1 and 0, sensor nodes are forced to collaborate when detecting flocks. Hence, all subsequent experiments test for a given flock number and flock pattern radius how many patterns can be detected as communication range c shrinks from $\frac{c}{p} = 1$ to $\frac{c}{p} = 0$. In order to maintain comparability with Laube *et al.* (2008) we ran the experiments with a flock contiguity $k = 3$ out of $|T| = 100$ time steps in total.

As `DDIG` claims to exploit deferred processing to achieve higher reliability, a first set of experiments varied the length of that deferral. Flocks were detected with deferrals ranging from $d = 0$ to $d = 9$, which is three times the flock contiguity $k = 3$ (Figure 8). The selection of the furthermore required parameters comb width $w = 1.2$ and partial pattern size $n' = 8$ will be discussed in section 5.4.

Results. Figure 8 illustrates the effect of deferred processing on *eoo*. The signature *eoo* response curve is of sigmoid shape, expressing little to no missed patterns around $\frac{c}{p} = 1$, then showing a transition phase where performance drops and *eoo* increases, and finally complete failure with very small communication ranges c around $\frac{c}{p} = 0$. This failure occurs as c decreases to a level where the network becomes disconnected. At these levels, the inter-node communication upon which the `DDIG` algorithm depends becomes impossible. An *eoo*=0.2 means that out of $100 * 5 = 500$ patterns a total of 400 were correctly detected and 100 were missed.

Starting off with zero deferral, the sigmoid curve for $d = 0$ shows a significant performance drop as early as 0.6. Each deferral extension added to this benchmark moves the curve towards the left, expressing improved performance. This effect, however, seems to thin out with prolonged deferrals. After a deferral of about 6, that is twice the flock contiguity $k = 3$, the impact of any added deferral does not seem to significantly improve the performance anymore. This suggests that there is a cut-off deferral where the grazing potential is exhausted. Attempting to optimize the gains from deferral and the risk of out-dated findings, all subsequent experiments were conducted with a deferral $d = 3$ (solid black signatures in figure 8).

With the chosen parameters no *eoc* occurred, hence no graph is provided for *eoc* in experiment #1.

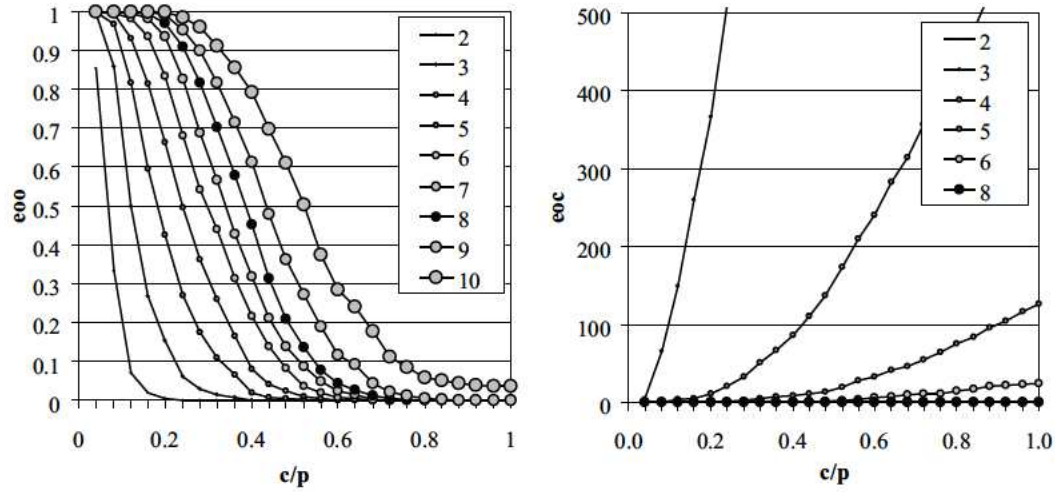

Figure 9. Partial patterns. The two figures illustrate DDIG's capability for balancing *eoo* and *eoc*. (left) Results are improved when the task difficulty is relaxed from searching complete flocks of size $n = 10$' to partial flocks of size $n' = 2$. (right) The such achieved reduction of *eoo* is, however, bought through emerging *eoc* for very small partial patterns, *eoc* is indicated in absolute numbers of false positives. Both series used $d = 3$ and $c = 1.2$. The size of the partial flock is coded in signature size, the solid black signature refers to the corresponding curve for $d = 3$ and $n' = 8$ in Figure 8.

5.4. *Experiment #2: Partial patterns*

Decentralized spatial computing aspires to complete the same tasks as its centralized counterparts. In some cases this may be difficult since the decentralized nodes have only access to partial knowledge about the problem at hand. In such limiting conditions it can be adequate to allow the algorithm to produce estimation or approximation results, that is results of a slightly lower information content, but still sufficient in many application contexts. In the case of DDIG, the pattern mining task becomes increasingly difficult when the communication range to pattern size ratio $\frac{c}{p}$ becomes very small. This second set of experiments illustrates how DDIG can still produce useful knowledge – admittedly imperfect knowledge – under increasingly difficult conditions.

DDIG is tested for a relaxation of the crisp task of 'find patterns of size $n = 10$' to 'also report partial patterns of size $n' < 10$'. Lowering the difficulty by searching for partial patterns comes for the prize of detection error. For example, reporting a flock ($n = 10$) when only a partial flock of size $n' = 6$ is detected, reduces the danger of missing flocks (error of omission, *eoo*), for the price of the odd false positive (error of commission *eoc*). In some application contexts, however, certain forms of detection error are tolerable and then it is the user's responsibility to find an acceptable balance for *eoo* and *eoc*. The second set of experiments explored the relation of *eoo* and *eoc* for variable sizes of partial patterns (Figure 9).

Results. The purposeful relaxation of the task searching for partial flocks, results in significantly better pattern detection results (Figure 9, left). This holds especially true for the crucial $\frac{c}{p}$-interval below 0.5. For instance, searching a partial flock of size $n' = 8$ reduces *eoo* for $\frac{c}{p} = 0.4$ by 37%. As can be seen from the right graph, this relaxation does not result in any *eoc* with the used experiment parameters. Flocks of $n' = 8$ are still a very strong indicator for a flock of $n = 10$. As can be seen from the same graph, this picture completely changes when searching for very small partial patterns. For $n' = 4$ and below large numbers of false positive patterns are detected. Depending on the applications field, users of DDIG may opt for different balances of *eoo* and *eoc*. In the context of this paper $n' = 8$ was used

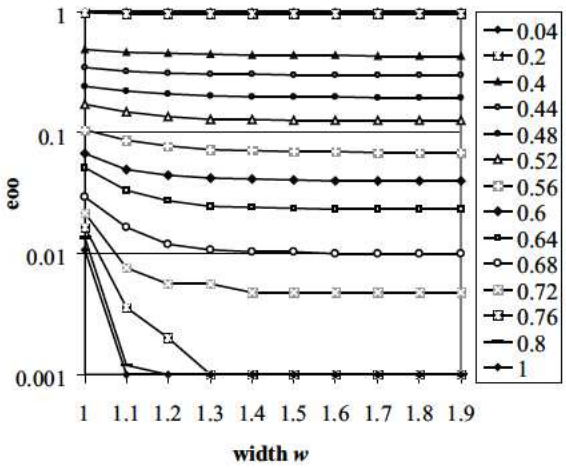

Figure 10. Selecting combing width w. From varying the combing width and keeping constant all other parameters revealed a point of inflection at 1.2. Since any further extension of this measure did not result in further improvement (straight curves to the right of the inflection point), a universal combing width $w = 1.2$ was chosen.

as it yielded reasonable results for no added *eoc*.

5.5. *Experiment #3: Combing width*

At a very early stage in the experimental procedure an initial set of experiments was used to define a reasonable combing width w required for the refine step. In this experiment the combing width was varied growing from $w = 1$ to $w = 1.9$ for various communication ranges c whilst deferral d and partial pattern size n' were kept constant (Figure 10).

Results. As can be seen for different communication ranges, a combing width of $w = 1.2$ ascertains the inclusion of all relevant position fixes in the combing stage and any further extension of w does not result in reduced *eoo* for the given parameterisation. For this reason and in order to allow for cross experiment comparisons a universal combing width of 1.2 was used in all previous experiments.

6. Evaluation with real observation data: Grazing cattle

In a second set of experiments `DDIG` was tested on trajectory data of tracked livestock grazing in confinement. After having achieved promising results with deferred decentralized information grazing in an artificial simulation environment, `DDIG` was put to the test with real animals showing real social behavior. For the sake of argument, the dictional faux pas of 'flocking cows' shall be excused within this paper[1].

6.1. *Data*

The trajectories emerged from a smart farming project aiming at applying WSN technology in agro-research, carried out with the Autonomous Systems Lab, CSIRO ICT Centre and Rendel Labs, CSIRO Livestock Industries. This cattle monitoring

[1]Even more so as the English language takes the liberty of letting shepherds tending flocks of sheep.

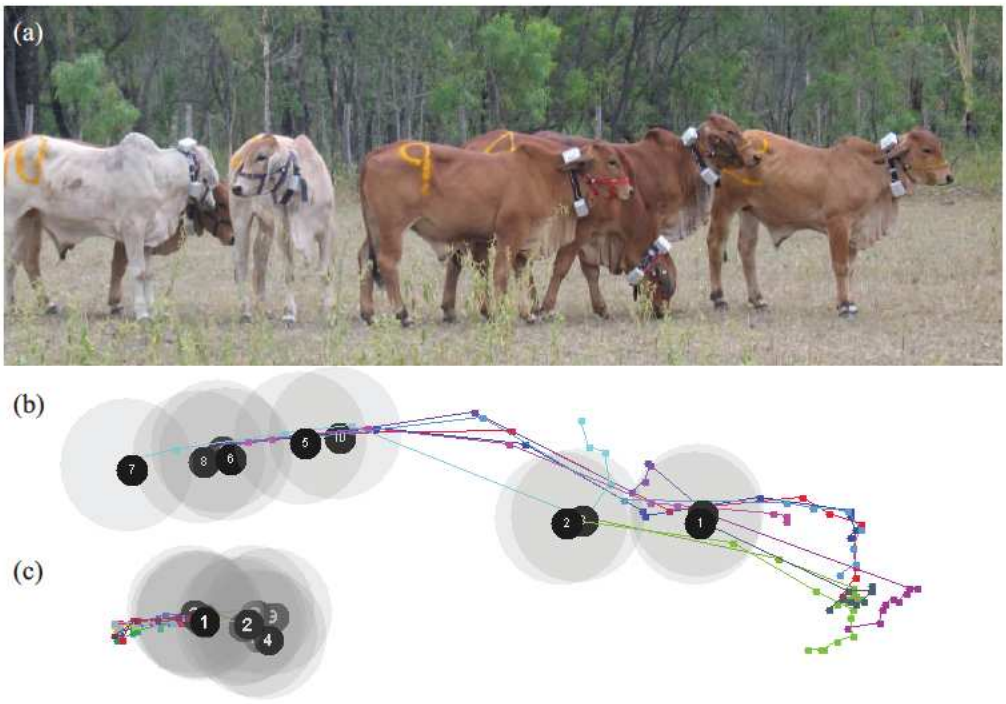

Figure 11. GPS tracked cattle. (a) Individual cows fitted with GPS collars. (b,c) Trajectories of $n = 10$ cows with a communication range $c = 25$m (grey discs) and memory tails of extent $e = 10$min. In (b) the group moves quickly to the left in an elongated file arrangement and hence does not satisfy the definition of a flock of radius $p = 25$m, the flock is passive. In (c), however, the cows slowly graze, heading to the right in an active flock formation (bold letters indicating individuals detecting a flock). Image source: Autonomous Systems Lab, CSIRO ICT Centre and Rendel Labs, CSIRO Livestock Industries

project investigates grazing cattle behavior in conjunction with environmental conditions change. Female cows (*Bos taurus*) were tagged with GPS data loggers and their trajectories monitored whilst the cows were grazing on a paddock of 600m times 200m at the CSIRO Rockhampton research site. A data set covering $n = 10$ individuals for an uninterrupted 24h interval was selected and the temporal granularity was resampled from too fine milliseconds to minutes, $|T| = 1440$ time steps in total (see Figure 11).

The 24h cycle includes extended time periods of resting. Since this paper is only interested in movement, these stationary phases were excluded in a preprocessing step. The group was only considered moving, when the distance of the consecutive minimal enclosing circle (MEC) was $d \geq 5m$. This resulted in a reduction to $|T_{move}| = 569$ time frames.

6.2. *Methodology and preliminaries*

Testing `DDIG`'s reliability on observation data is somewhat more difficult than within a fully controllable simulation scenario discussed in the previous section. The crux is that it is neither *a priori* defined when the cattle flock nor how such a flock should be parameterized. The following assumptions have been taken to still allow a critical evaluation of `DDIG`'s plausibility and reliability on the cattle trajectories.

As a first preliminary the flocking radius p was deduced from the trajectories. Following the spatial constraint of a circular pattern area in the flock definition, for each time step the MEC for all n cows was computed and averaged over all $|T_{move}| = 569$ time steps, resulting in an average flocking radius of $p = 24.71$m. All following experiments are carried out with a rounded $p = 25$m.

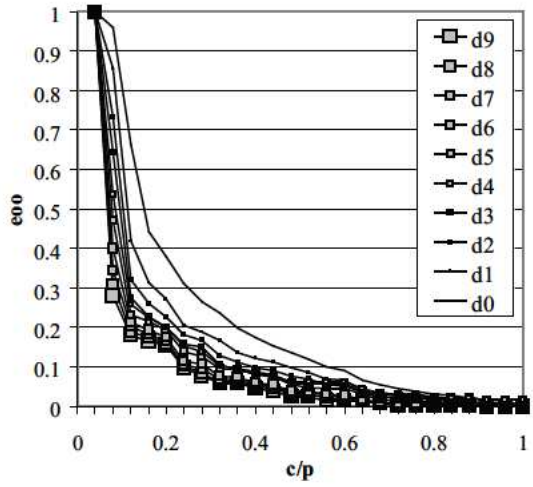

Figure 12. Deferred processing in the cattle experiments. Applied to trajectories of tracked cattle, DDIG again appears to successfully exploit the advantages of deferred processing, however to a lesser degree.

As a second assumption all $n = 10$ cows are assigned to one single flock. However, this flock can be active or passive. *Active* refers to times when all $n = 10$ cows were located within a flocking radius p, *passive* otherwise. Out of $|T_{move}| = 569$ a total of 287 were active flock constellations, 282 passive (non-flocking). Note that this contrasts to CRW simulations in the previous section where all implanted flocks were permanently active throughout the entire experiment.

As with the simulated data, *eoo*- and *eoc*-detection error was evaluated in a number of experiments for variable $\frac{c}{p} - ratios$. In all the experiments, the sensor nodes search for flocks using DDIG and then compare their decentralized finding with the globally computed reference (active vs. passive constellation). In order to allow for comparison with section 5, all subsequent experiments were carried out for $k = 3$ flock contiguities, corresponding to 3 minutes flocking.

6.3. *Experiment #4: Deferred processing*

Just as with the CRW simulated trajectories, the first experiment investigated the potential of deferred processing. In contrast to the simulated sensor nodes, real cows were expected to show less predictable and also less regular roaming behavior. The first question was hence whether or not, and to what extent deferred processing would decrease detection error *eoo*. Again, flocks were detected for deferrals ranging from $d = 0$ to $d = 9$ and ratios $\frac{c}{p} = 1$ to $\frac{c}{p} = 0$.

Results. First, it can be observed that also with the real tracking data, larger deferrals resulted in fewer missed patterns, that is smaller *eoo*. However, as a second observation it must be admitted that improvement achieved through deferred processing seemed to be less distinctive than with the CRW simulated data. As far down as $\frac{c}{p} = 0.2$ the algorithm detected across all deferrals as much as 80% of all flocking situations. In this range, added deferral resulted in little improvement, hardly more than 10% improvement from $d = 0$ to $d = 9$. When comparing Figure 8 with Figure 12 a third observation relates to the general shape of the error curves. In contrast to the CRW trajectories, *eoo* for the cattle data is kept low for much smaller $\frac{c}{p}$-ratios. In the latter case the algorithm suddenly and dramatically fails around $\frac{c}{p} = 0.1$.

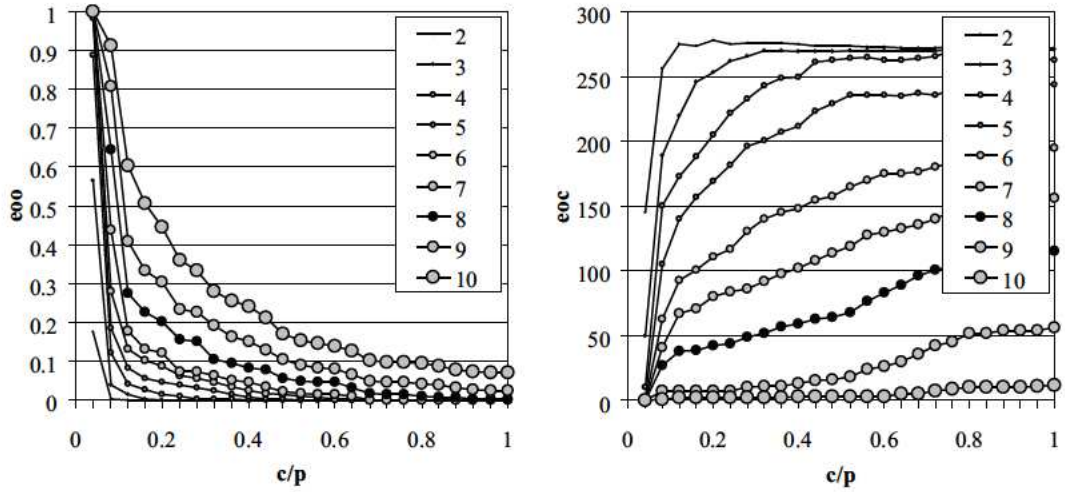

Figure 13. Partial pattern mining in the cattle experiments. Paying tribute to imperfect information and relaxing the task to finding partial flocks results also in the cattle experiment in less error. However, anything beyond $n' = 8$ results in large *eoc* for too little gain in *eoo*.

6.4. *Experiment #5: Partial patterns*

The second set of experiments investigated the effect of relaxing the task from finding crisp flocks to finding partial flocks. This experiment evaluated DDIG's potential for balancing *eoo* and *eoc* when searching for imperfect knowledge for the cattle case.

Note that there is a difference in computing *eoc* ('false positives') between the CRW simulated trajectories and the cattle tracking data. Whereas in the first case the $A - (5n) = 150$ non-flocking sensor nodes may contribute in large numbers of false positives concurrent to the $5n = 50$ flocking sensor nodes, in the latter case there are no additional non-flocking cows. Hence, for every considered time step there are four possible outcomes: (1) there is an active flock and it is correctly detected, (2) there is no active flock (passive flock) and none is detected, (3) there is an active flock and it is missed (*eoo*), and (4) there is no active flock but the algorithm wrongly infers a flock (*eoc*). In this experimental setup, not only *eoo* is limited as in section 5 (i.e. the aforementioned 287 active flock constellations), but also *eoc* has a maximal value. This eoc_{max} results when a flock is wrongly detected for all remaining 282 passive constellations, that is non-flocking constellations out of 569 totally evaluated ($287 + 282 = 569$).

Results. Comparing the respective graphs in Figure 9 with Figure 13, again the stricter and more difficult the task (large n'), the higher both *eoo* and *eoc*. Again, the curves are rather hyperbolic in shape. *eoo* and *eoc* almost complement each other, bad performance in one graph (e.g. *eoo* for $n = 10$) corresponds to a good performance in the other (e.g. *eoc* for $n = 10$), and vice versa ($n' = 2$). Furthermore, it appears that partial flocks below $n' = 8$ do not significantly reduce *eoc* (left), but dramatically increase *eoc* (right). Hence, balancing *eoo* and *eoc* is most effective for large partial flocks between $n' = 8$ and $n = 10$. As expected, *eoc* plateaus around eoc_{max} for small partial flocks.

7. Discussion

A first key finding from both experiment series is that increasing the deferral results in better pattern detection, that is in less *eoo* for acceptable *eoc*. The results reveal that mobility indeed can be exploited for decentralized spatial computing when deferred processing is acceptable. This finding is in accordance with similar findings in current WSN research, where various computational advances could be identified for lagged or even delay-tolerant information routing (Grossglauser and Tse 2002, Grossglauser and Vetterli 2006). Just as for some routing applications a late message is perfectly acceptable and more valuable than no message at all, also for decentralized movement pattern detection a deferred pattern notification is in many contexts sufficient.

As a second finding it can be noted that the performance enhancing effect of deferred processing is biggest for the most current past and fades out when allowing for increasing deferral. This finding is reasonable as it has to be expected that after a while the potential from exchanging and aggregating tails is exhausted and any further deferral can not result in more information. As deferred processing inevitably derogates the value of the finding and longer deferrals require larger memories and more information processing, for any real application an optimal deferrral has to be found. The optimal length of the deferral depends on the movement regime of the sensor agents, the more they 'wiggle', the more they can exchange tails. However, with both the chosen CRW regime and the real cattle data a deferral d in the magnitude of the flock contiguity k produced good results.

Thirdly, our experiments raise the question whether one could step away from aiming at perfect knowledge in decentralized data mining, but adjust the task to less absolute but more achievable goals. It has been widely acknowledged that detecting (n, k, p)-*flock* is difficult already in a static setting (Gudmundsson *et al.* 2007) and even more so in a dynamic system (Laube *et al.* 2008). As has been shown in experiments #2 and #5, `DDIG` struggles in detecting crisp (n, k, p)-*flock* for small communication ranges c. If this ambitious goal is relaxed to slightly smaller parital (n', k, p)-*flock*s the results rapidly improve. As was also illustrated in the experiments, such approximation approaches may result in detection error. However, trading *eoo* for *eoc* offers a strategy for balancing such error risks in a given application scenario. One should not confuse the notion of partial flocks with probability, as finding a partial flock of size $n' = 8$ allows no probabilistic inference on the presence of a flock of cardinality $n = 10$. However, it is a strong indication for the presence of a large flock and this information is with all its uncertainties and limitations potentially useful in many application contexts.

Only limited direct comparisons are possible between the experiments with random walk trajectories and the real cattle trajectories. Whereas the CRW results mirror an average of several simulation runs each in smooth curves, the one and only cattle data set results in rough edges and peculiarities. Nevertheless, it is fair to say that in both cases deferred processing resulted in reduced detection error. With the smoother and less 'wiggly' cattle trajectories the data exchange potential is limited and hence deferred processing results in less detection error reduction. Furthermore, the exclusion of stationary intervals makes the pattern detection task harder, potentially resulting in less improvement through deferred processing.

In contrast to the decentralized flock detector `FLAGS` presented in Laube *et al.* (2008), the `DDIG` algorithm discussed here requires localized sensor nodes. Constant localization has obvious advantages but comes at a potentially high price in terms of battery life, weight, and hardware costs. The advantage of localized sensor nodes lies in the potential of memorizing, exchanging and ideally synchronizing

the knowledge about the neigbors' past locations. The qualitative neighbor lists available in FLAGS are of a much lower information content and their aggregation has to remain less powerful. At the same time, an algorithm relying on localization is vulnerable to uncertainties inherent to any localization technology.

Both experiment series underline that there are limitations to decentralization for movement pattern mining. In both cases the *eoo* curves showed rapid failure for small communication ranges. When very small communication ranges c are chosen, sensor nodes become disconnected and hence unable to collaborate. This is, however, not a weakness of the presented algorithm but rather a feature of any wireless network relying on some degree of connectivity. Any forced performance enhancement beyond that natural basic connectivity threshold would be guessing and hence be inappropriate.

8. Conclusions and Outlook

In this paper we investigated the concept of decentralized spatial computing (DeSC) for in-network data mining in mobile wireless sensor networks. We presented the algorithm DDIG (Deferred Decentralized Information Grazing) allowing collaborating roaming sensor nodes to detect the movement pattern *flock* directly in the network, that is locally and without the need for conventional data integration in a global omniscient database. DDIG achieves this task enabling the moving sensor nodes to graze location information whilst moving, hence sensor nodes compensate for their limited spatial perception range through extending their temporal perception range. DDIG is based on a combination of a basic in-node memory, local exchange and aggregation of grazed location information, as well as deferred processing and an adaptation of *filter-refine* in the actual data mining task. We tested our algorithm not only with simulated random walk trajectories but also put it to the test with grazing cattle showing 'real' movement behavior. In both scenarios DDIG performed better with increasing deferrals.

DDIG illustrates that the constantly changing topology in a mobile wireless sensor network can be exploited for decentralized spatial computing. Allowing roaming sensor nodes to first graze information over time and then allowing for a deferral of the actual information processing proved to be a valuable strategy for aggregating spatiotemporal information in a dynamic geosensor system. In a decentralized in-network solution all communication and collaboration must remain local, DDIG is highly scalable and allowed implementing a straightforward adaptation of filter-refine without the need of difficult to maintain global index structures. Finally, the experiments confirm earlier findings on decentralized spatial computing in that the best results were achieved when allowing some error tolerance, that is relaxing the task from seeking perfect knowledge to getting imperfect but still useful knowledge.

Future work will put more effort in the notion of pattern probabilities, aiming at providing evidence for the presence of movement patterns tied to some form of probability. Instead of aiming for binary indications of the presence of absence of a pattern, future decentralized movement pattern detectors should allow sensor nodes to issue statements such as 'node a detected a (n, k, p)-*flock* with a probability of 0.8'. Such detectors could very much play the role of probing procedures conditionally inducing more costly but more inclusive testers. Both the simulated CRW and the observed cattle trajectory data proved to be rich repositories for analyzing complex group behavior. Such data is open to movement analysis beyond the flocking pattern and the DeSC problem discussed in this paper. Alternative movement patterns of interest could be leadership or convergence and divergence. In terms of further testing, we plan to implement DDIG in a local sensor lab fea-

turing 100 nodes. Experiments with 'flocking students' in a controllable campus setting will without doubt reveal implementation issues relevant for fine-tuning algorithms developed in a simulation context. Finally, measurement errors are inherent to any real geosensor system. For the development of `DDIG` we excluded issues of imperfection for simplicity reasons, the CRW trajectories were simulated without measurement error, in the cattle case we ignored potentially present measurement error. However, the implementation of `DDIG` and similar algorithms in real world geosensor systems will have to include error handling strategies.

Achnowledgements

Patrick Laube's work was funded by the Australian Research Council (ARC), Discovery grant DP0662906 and the ARC Research Network on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP). Matt Duckham's research was supported by ARC Discovery Grant DP0662906. The authors furthermore thank Tim Wark, Autonomous Systems Laboratory, CSIRO ICT Centre, Pullenvale QLD and Rendel Labs, CSIRO Livestock Industries, Australia for providing the excellent data set and valuable comments.

References

Andersson, M., Gudmundsson, J., Laube, P. and Wolle, T., 2008. Reporting Leaders and Followers among Trajectories of Moving Point Objects. *GeoInformatica*, 12 (4), 497–528.

Becker, L., Giesen, A., Hinrichs, K.H. and Vahrenhold, J., 1999. Algorithms for performing polygonal map overlay and spatial join on massive data sets. *In*: R. Güting, D. Papadias and F. Lochovsky, eds. *Advances in Spatial Databases.*, Vol. 1651 of *Lecture Notes in Computer Science* Heidelberg: Springer, 270–285.

Benkert, M., Gudmundsson, J., Hübner, F. and Wolle, T., 2008. Reporting flock patterns. *Computational Geometry*, 41 (3), 111–125.

Betteridge, K., 2008. Monitoring urine distribution by grazing livestock. *In*: The Organizing Committee of the IGC/IRC Congress, ed. *International IGC-IRC Congress*. Hohhot, China: Guangdong People's Publishing House.

Braginsky, D. and Estrin, D., 2002. Rumor routing algorthim for sensor networks. *In*: C.S. Raghavendra and K.M. Sivalingam, eds. *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*. Atlanta, GA: ACM Press, 22–31.

Brinkhoff, T., Kriegel, H.P., Schneider, R. and Seeger, B., 1994. Multi-step processing of spatial joins. *SIGMOD Rec.*, 23 (2), 197–208.

Chatterjea, S., S., K. and P, H., 2006. Sensor Networks. *GEOconnexion International Magazine*, 20–22.

Cheng, Z. and Heinzelman, W.B., 2005. Flooding strategy for target discovery in wireless networks. *Wireless Networks*, 11, 607–618.

Duckham, M., Nittel, S. and Worboys, M., 2005. Monitoring dynamic spatial fields using responsive geosensor networks. *In*: C. Shahabi and O. Boucelma, eds. *13th annual ACM international workshop on Geographic Information Systems*. Bremen, Germany: ACM Press, 51–60.

Esperanca, C. and Samet, H., 1997. Orthogonal polygons as bounding structures in filter-refine query processing strategies. *In*: G. Goos, J. Hartmanis and J. van Leeuwen, eds. *Advances in Spatial Databases.*, Vol. 1262 of *Lecture Notes in Computer Science* Heidelberg: Springer, 197–220.

Estrin, D., Govindan, R. and Heidemann, J., 2000. Embedding the Internet - Introduction. *Communications of the ACM*, 43 (5), 38–41.

Farah, C., Zhong, C., Worboys, M. and Nittel, S., 2008. Detecting Topological Change Using a Wireless Sensor Network. *In*: T.J. Cova, K. Beard, M.F. Goodchild and A.U. Frank, eds. *GIScience 2008*, Vol. 5266 of *LNCS* Heidelberg: Springer, 55–69.

Grossglauser, M. and Tse, D.N.C., 2002. Mobility increases the capacity of ad hoc wireless networks. *Ieee-Acm Transactions on Networking*, 10 (4), 477–486.

Grossglauser, M. and Vetterli, M., 2006. Locating mobile nodes with EASE: learning efficient routes from encounter histories alone. *Networking, IEEE/ACM Transactions on*, 14 (3), 457–469.

Gudmundsson, J., van Kreveld, M. and Speckmann, B., 2007. Efficient Detection of Patterns in 2D Trajectories of Moving Points. *GeoInformatica*, 11 (2), 195–215.

Karp, B. and Kung, H.T., 2000. GPSR: Greedy perimeter stateless routing for wireless networks. *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*. Boston, MA: ACM Press, 243–254.

Kellerer, W., Bettstetter, C., Schwingenschlogl, C., Sties, P. and Steinberg, K.E., 2001. (Auto) mobile communication in a heterogeneous and converged world. *IEEE Personal Communications*, 8 (6), 41–47.

Laube, P. and Duckham, M., 2009. Decentralized spatial data mining for geosensor networks. *In*: H. Miller and J. Han, eds. *Geographic data mining and knowledge discovery*. 2nd, in press, ed. London: Taylor and Francis.

Laube, P., Duckham, M. and Wolle, T., 2008. Decentralized Movement Pattern Detection amongst Mobile Geosensor Nodes. *In*: T.J. Cova, K. Beard, M.F. Goodchild and A.U. Frank, eds. *GIScience 2008*, Vol. 5266 of *LNCS* Heidelberg: Springer, 199–216.

Laube, P. and Imfeld, S., 2002. Analyzing Relative Motion within Groups of Trackable Moving Point Objects. *In*: M.J. Egenhofer and D.M. Mark, eds. *Geographic Information Science.*, Vol. 2478 of *Lecture Notes in Computer Science* Heidelberg: Springer, 132–144.

Laube, P., van Kreveld, M. and Imfeld, S., 2004. Finding REMO - Detecting Relative Motion Patterns in Geospatial Lifelines. Proceedings of the 11th International Symposium on Spatial Data Handling, *In*: P.F. Fisher, ed. *Developments in Spatial Data Handling*. Heidelberg: Springer, 201–214.

Lynch, N., 1996. *Distributed Algorithms*. San Mateo, CA: Morgan Kaufmann.

Nittel, S., Stefanidis, A., Cruz, I., Egenhofer, M.J., Goldin, D., Howard, A., Labrinidis, A., Madden, S., Voisard, A. and Worboys, M., 2004. Report from the First Workshop on Geo Sensor Networks. *ACM SIGMOD Record*, 33 (1).

Sadek, M. and Duckham, M., 2008. Effect of Neighborhood on In-Network Processing in Sensor Networks. *In*: T.J. Cova, K. Beard, M.F. Goodchild and A.U. Frank, eds. *GIScience 2008.*, Vol. 5266 of *LNCS* Heidelberg: Springer, 133–150.

Wark, T., Corke, P., Sikka, P., Klingbeil, L., Guo, Y., Crossman, C., Valencia, P., Swain, D. and Bishop-Hurley, G., 2007. Transforming Agriculture through Pervasive Wireless Sensor Networks. *Pervasive Computing, IEEE*, 6 (2), 50–57.

Werner-Allen, G., Lorinez, K., Ruiz, M., Marcillo, O., Johnson, J., Lees, J. and Welsh, M., 2006. Deploying a wireless sensor network on an active volcano. *IEEE Internet Computing*, 10 (2), 18–25.

Zhao, F. and Guibas, L.J., 2004. *Wireless Sensor Networks – An Information Processing Approach*. San Francisco, CA: Morgan Kaufmann Publishers.